UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/763,777	01/22/2004	Anuj B. Gosalia	MSFT-2857/304862.02'	7562

41505 7590 11/30/2007

WOODCOCK WASHBURN LLP (MICROSOFT CORPORATION)
CIRA CENTRE, 12TH FLOOR
2929 ARCH STREET
PHILADELPHIA, PA 19104-2891

EXAMINER	
LEE, KWOK W	

ART UNIT	PAPER NUMBER
4113	

MAIL DATE	DELIVERY MODE
11/30/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/763,777	GOSALIA ET AL.	
	Examiner	Art Unit	
	Kwok Wing Lee	4113	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 March 2004 and 23 February 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *15-51 and 66-73* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *15-51 and 66-73* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *22 January 2004 and 23 February 2007* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *01/22/04 and 06/22/07*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Election/Restrictions

Applicant's election without traverse of group 1 consisting of claims 1-73 in the 11/13/2007 communication is acknowledged. Additionally, the cancellation of claims 1-14, 52-65 and 74-79 has also been acknowledged.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 26 and 36 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. A "modulated data signal" is not statutory subject matter because it does not belong to one of the statutory categories (See MPEP 2106.01). It is not a "process, machine, manufacture, or composition of matter".

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

Claims 15-18, 25-27, 41-43, 66-67, 69 and 71-72 are rejected under 35 U.S.C. 102(b) as being anticipated by Notenboom et al (US 5,748,468).

With respect to claim 15, Notenboom teaches a method for scheduling tasks for processing by a coprocessor, comprising: gathering tasks for processing by a coprocessor (Co-processor 44, see figure 1) into a memory group (Memory System 30, see figure 1) wherein the memory group relates to a first application (Column 3, lines 25-36); delivering the tasks to a scheduler (Task Scheduler 142, see figure 5) wherein scheduler functions include determining an order for processing the tasks wherein the order may include tasks that relate to one or more other applications (Column 3, lines 13-24); determining an order for processing the tasks wherein the order accounts for any relative priority among the first application and one or more other applications and a corresponding amount of processing time that the first application and one or more other applications are entitled to (Column 3, lines 36-42); preparing tasks for processing by ensuring that any needed memory resources are available in a coprocessor-accessible memory location wherein the preparing tasks occurs in the order determined by the scheduler (Column 8, lines 61-65); and submitting tasks prepared according to the preparing to the coprocessor for processing (Column 3, lines 13-24).

With respect to claim 16, Notenboom teaches a method according to claim 15 wherein the coprocessor includes a graphics processing unit (GPU) (Co-processor 44 in figure 1 referenced in column 7, lines 7-15 and as explained in column 1, lines 39-42, a DSP that processes video signals is inherently a graphics processing unit).

With respect to claim 17, Notenboom teaches a method according to claim 15, further comprising calling an Application Program Interface (API) when the first

application has one or more tasks that require processing by the coprocessor (Column 7, lines 7-15).

With respect to claim 18, Notenboom teaches a method according to claim 17, further comprising calling a user mode driver wherein the functions of the user mode driver include placing rendering commands associated with the one or more tasks in the memory group (Column 6, line 62-column 7, line 3, it is inherent that the multi-tasking Windows operating system has a user mode or space where graphic related commands are placed into memory).

With respect to claims 25-27, see above discussion regarding claim 15.

With respect to claim 41, see above discussion regarding claim 15.

With respect to claim 42, see above discussion regarding claim 16.

With respect to claim 43, Notenboom teaches an apparatus according to claim 41 wherein the coprocessor supports interruption (Column 1, lines 52-56) during the processing of a task by automatically saving task information to a coprocessor-accessible memory location. It is an inherent teaching that concurrent multitasking involves the interruption and saving of context information to be loaded again later; multitasking does not involve actual concurrent running tasks but merely the context switching of tasks to seemingly operate as if they were concurrently running.

With respect to claim 66, Notenboom teaches a coprocessor for use in connection with a co-processing scheduler, comprising: a coprocessor (Co-Processor 44, see figure 1) for processing tasks that are submitted to the coprocessor by a scheduler process (Column 3, lines 13-23) wherein the scheduler process submits tasks

to the coprocessor according to a priority of applications that request processing of the tasks (Column 3, lines 13-15), and wherein the priority determines the amount of coprocessor time one or more applications are entitled to (Column 3, lines 36-42).

With respect to claim 67, Notemboom teaches a coprocessor according to claim 66 wherein the tasks are first stored in an application-specific memory location (Column 8, lines 20-32).

With respect to claim 69, Notemboom teaches a coprocessor according to claim 66 wherein the coprocessor processes tasks from a run list by switching immediately to a subsequent task on the run list when a switching event occurs (Column 7, lines 65-67 and column 8, lines 20-32, it is inherent that this switching is immediate).

With respect to claim 71, Notenboom teaches a coprocessor according to claim 66 wherein the coprocessor comprises a GPU (Co-processor 44 in figure 1 referenced in column 7, lines 7-15 and as explained in column 1, lines 39-42, a DSP that processes video signals is inherently a graphics processing unit).

With respect to claim 72, Notenboom teaches a coprocessor according to claim 66 wherein the coprocessor accesses memory resources in a coprocessor-readable memory by a memory manager (Column 3, lines 13-24 and column 6, lines 18-29).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the

invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 19 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Notenboom et al (US 5,748,468) in view of well-known practice in the art.

With respect to claim 19, all of the limitations of claims 15, 17 and 18 have been addressed above. Notenboom does not teach a method further comprising returning the rendering commands to the API, and submitting them to a coprocessor kernel. However, at column 6, line 62-column 7, line 3, states the use of a multi-tasking Windows operating system. There are multi-tasking Windows operating systems with kernel modes, and official notice of such is taken. A kernel mode has full access to memory and controls the carrying out of tasks to be performed by a processor. It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the host operating system used in Notenboom to return rendering commands to an API and submitting these commands to the kernel of the operating system controlling the coprocessor.

With respect to claim 20, all of the limitations of claim 15 have been addressed above. Notenboom teaches an operating system for loading tasks to be processed by a coprocessor. Notenboom does not teach a method further comprising generating a Dynamic Memory Access (DMA) buffer by a kernel mode driver wherein one or more tasks that require processing by the coprocessor are used to generate the DMA buffer, and the DMA buffer represents the one or more tasks used to generate the DMA buffer. However, the multi-tasking Windows operating system used in Notenboom has a kernel mode that has full access and control of system memory, as addressed above. A DMA

buffer is well known in the art. It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the operating system for loading tasks to the coprocessor to have the kernel mode of the operating system to create a DMA buffer (if not already present) for tasks that require processing because a DMA buffer is a form of memory and is a well-known type of memory in the art.

Claims 21-23, 28-37, 44-51, 68, 70 and 73 are rejected under 35 U.S.C. 103(a) as being unpatentable over Notenboom et al (US 5,748,468) in view of Moore et al (US 5,437,017).

With respect to claim 21, all of the limitations of claims 15 and 20 have been addressed above. Notenboom does not teach a method further comprising generating a list of memory resources by the kernel mode driver wherein the memory resources represented by the list are needed by the coprocessor to process one or more tasks represented by the DMA buffer. The Moore reference teaches a multitasking multiprocessor system using paging with a fault scheme where missing memory resources are listed as an index (Column 7, lines 44-57). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom with Moore's multitasking system to have Notenboom's operating system's kernel mode to list needed memory resources as suggested by Moore (Column 7, lines 44-57),

where a DMA buffer is a well-known type of memory and would have been also obvious to use as representing one or more tasks.

With respect to claim 22, all of the limitations of claims 15, 20 and 21 have been addressed above. Notenboom does not teach a method further comprising building a paging buffer for bringing the memory resources on the list of memory resources to correct memory addresses within the coprocessor-accessible memory location. The Moore reference teaches a multitasking system where a translation lookaside buffer is used for paging of tasks to memory (Column 2, line 62-column 3, line 27). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multitasking multiprocessor of Notenboom with the multitasking system of Moore, as described above, to further include a translation lookaside buffer to translate needed resources from virtual address space to correct physical memory locations because it is a well-known technique in the art; physical memory used to load applications on a computer is limited and this limitation is solved by loading other applications in virtual memory, typically using memory from a hard drive, to act as physical memory.

With respect to claim 23, all of the limitations of claim 15 have been addressed above. Notenboom does not teach a method wherein said preparing is accomplished by a preparation thread which calls a memory manager process capable of determining a location in the coprocessor-accessible memory location to page any needed memory resources. The Moore reference teaches a multitasking system where paging of memory is used (Column 2, line 62-column 3, line 27). It would have been obvious at

the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multitasking multiprocessor system of Notenboom with the multitasking system of Moore, as described above, further having a preparation thread to call a memory manager process in the operating system capable of determining paging locations for needed memory resources; a kernel in an operating system that controls and allocates memory for paging is well-known in the art; any action is initiated by a thread of execution and is also well-known in the art.

With respect to claim 28, lines 1-15 of the claim have been addressed above in claim 15. Notenboom teaches managing the coprocessor-readable memory to apportion the coprocessor-readable memory among the various tasks (Column 2, lines 1-5). Notenboom does not teach providing a virtual address space for the tasks. The Moore reference teaches a multiprocessor system that executes multiple tasks simultaneously, where each of these multiple tasks has a virtual address space within memory (Column 1, lines 52-59). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multitasking multiprocessor system of Notenboom to have configured its operating system and graphics processing unit to executing multiple applications and provide each executing application with its own virtual address space into memory because virtual addressing is a well-known method used in paging.

With respect to claim 29, all of the limitations of claim 28 have been addressed above. Notenboom teaches a method according to 28 wherein the coprocessor is a graphics processing unit (GPU) (Co-processor 44 in figure 1 referenced in column 7,

lines 7-15 and as explained in column 1, lines 39-42, a DSP that processes video signals is inherently a graphics processing unit).

With respect to claim 30, all of the limitations of claim 28 have been addressed above. Notenboom teaches a multiprocessor system using an operating system (Column 1, 52-56) and it is inherent that a user mode driver is provided because the operating system has a user mode; an operating system is also responsible for task and memory management. Notenboom does not teach a method further comprising storing a task in a DMA buffer wherein the storing is accomplished by a user mode driver. It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the system as taught by Notenboom to have stored task in a DMA buffer by a user mode driver since a DMA buffer is a well-known type of memory.

With respect to claim 31, all of the limitations of claims 28 and 30 have been addressed above. Notenboom does not teach a method further comprising validating a memory resource referenced in a resource list that is associated with the DMA buffer wherein validating entails finding a range of coprocessor-readable memory that is free and asking the kernel mode driver to map a page table or a memory resource handle to that range. The Moore reference teaches a multitasking multiprocessor system implementing paging (Column 2, line 62-column 3, line 27). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom and implement paging as used by Moore; whereby the operating system of

Notenboom, as addressed earlier, has a kernel mode, having full access and control of memory, to allocate free memory to be mapped by paging; as discussed above, a DMA buffer is a well-known type of memory in the art, as well as the paging technique described by claim 31.

With respect to claim 32, all of the limitations of claim 28 have been addressed above. Notenboom does not teach a method wherein the virtual address space is virtualized through the use of a flat page table that divides coprocessor-readable memory into pages of a predefined memory amount wherein further a page table is provided in the virtual address space that contains identifiers for specifying coprocessor-readable memory addresses. The Moore reference teaches a translation lookaside buffer (TLB) and a flat page table (Column 6, lines 34-40) where a virtual address is a search key that returns a physical address. It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the system of Notenboom with Moore, as described above, where paging further includes the use of a TLB and flat page tables because TLB and is well-known in the art with respect to paging and a flat page table is a well-known type of page table.

With respect to claim 33, see above discussions of claims 32, where the only difference is using a multi-level page table. A modification to use multi-level page tables would have been obvious because it is a well-known type of page table.

With respect to claim 34, all of the limitations of claim 28 have been addressed above. Notenboom teaches a method according to claim 28 wherein a portion of

coprocessor readable memory is used to indicate whether all required memory resources associated with a task that requires processing are available in coprocessor-readable memory (Column 8, lines 61-65).

With respect to claims 35-37, see above discussion regarding claim 28.

With respect to claim 44, all of the limitations of claims 41 and 43 have been addressed above. Notenboom teaches an operating system used in the concurrent multitasking multiprocessor system (Column 1, lines 52-56), which inherently teaches a private piece of coprocessor-accessible memory where a hardware state is saved when a task is not being processed. Notenboom does not teach an apparatus further comprising at least one of a private address space for one or more tasks, a private ring buffer where tasks are accumulated. The Moore reference teaches a multiprocessor system that executes multiple tasks simultaneously, where each of these multiple tasks has a virtual address space within memory (Column 1, lines 52-59). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multitasking multiprocessor system of Notenboom to have configured its operating system and graphics processing unit to executing multiple applications and provide each executing application with its own virtual address space into memory, as taught by Moore, because it is a well-known technique in the art; Notenboom's operating system inherently teaches that tasks are usually queued in a buffer and a ring buffer is a well-known type of buffer and would have also been obvious to use in the operating system.

With respect to claim 45, all of the limitations of claim 41 have been addressed above. Notenboom teaches an apparatus according to claim 41 wherein the coprocessor is capable of storing information regarding the history of coprocessor switches from task to task in a specified system memory location readable by the scheduler process (Column 1, lines 52-56, it is inherent that a coprocessor stores history information of task switches in a particular part of memory in order for the scheduler to determine which task to switch to next and to continue running where it left off).

With respect to claim 46, all of the limitations of claims 41 and 45 have been addressed above. Notenboom teaches an apparatus according to claim 45 wherein the coprocessor specifies a base address for the system memory location prior to storing information regarding the history of coprocessor switches from task to task in the system memory location (Column 1, lines 52-56, it is inherent that a coprocessor must specify a base address for before storing any information in order to have a location to store to).

With respect to claim 47, all of the limitations of claims 41 and 45 have been addressed above. Notenboom teaches an apparatus according to claim 45 wherein the coprocessor specifies a size for the system memory location prior to storing information regarding the history of coprocessor switches from task to task in the system memory location (Column 1, lines 52-56, it is inherent that a coprocessor must specify a size before storing any information in order to determine if there is enough memory to allocate for it).

With respect to claim 48, all of the limitations of claims 41 and 45 have been addressed above. Notenboom teaches an apparatus according to claim 45 wherein the coprocessor specifies a write pointer for indicating where in the system memory location the coprocessor should write to next (Column 6, lines 6-29).

With respect to claim 49, all of the limitations of claim 41 have been addressed above. Notenboom teaches an apparatus according to claim 41 wherein the coprocessor supports fence instructions that cause the coprocessor to write a piece of data associated with a fence instruction at an address specified in the fence instruction (Column 3, lines 57-67).

With respect to claim 50, all of the limitations of claim 41 have been addressed above. Notenboom teaches an apparatus according to claim 41 wherein the coprocessor supports trap instructions that are capable of generating a CPU interrupt when processed by the coprocessor (Column 3, lines 57-67).

With respect to claim 51, all of the limitations of claim 41 have been addressed above. Notenboom teaches an apparatus according to claim 41 wherein the coprocessor supports enable/disable context switching instructions such that when context switching is disabled, the coprocessor will not switch away from a current coprocessor task (Column 3, lines 57-67).

With respect to claim 68, all of the limitations of claim 66 have been addressed above. Notenboom does not teach a coprocessor wherein the coprocessor stores information related to a task in a per-context address space, and wherein further the information related to a task allows the coprocessor to process the task or a portion of

the task after processing one or more intervening tasks. The Moore reference teaches a multitasking multiprocessor that provides each task with a virtual address space (Column 1, lines 52-56). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom to use paging as shown by Moore so that each task can have their own virtual address space or the same as a per context address space were each task is switched in and out of processing from the virtual address space; paging and virtual addressing is a technique well-known in the art to provide extra memory to running applications supplementing physical memory.

With respect to claim 70, all of the limitations of claim 69 have been addressed above. Notenboom teaches a request by a central processing unto to switch to a new run list (Column 8, lines 10-32) and it is inherent that a switching event will comprise at least one of a completion of processing a previously submitted task because tasks are bound to be completed some time. Notenboom does not teach a coprocessor according to claim 69 wherein a switching event comprises a page fault in processing a task, a general protection fault in processing a task. The Moore reference teaches a page fault (Column 7, lines 44-57) and a general protection fault (Column 7, lines 7-18). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom with that of Moore, as described for claim 66, further comprising a general protection fault and a page fault during a switching event; faulting is a well-

know practice for error-checking in any computer system, where a general protection fault and a page fault are well-known types of faults.

With respect to claim 73, see above discussion regarding claim 68 and that memory resource accesses are inherently references to virtual memory addresses.

Claim 24 is rejected under 35 U.S.C. 103(a) as being unpatentable over Notenboom et al (US 5,748,468) and Moore et al (US 5,437,017) as applied to claims 15 and 23 and further in view of Fadavi-Ardekani et al (US 6,496,916).

With respect to claim 24, all of the limitations of claims 15 and 23 have been addressed above. Notenboom and Moore do not teach a method further comprising splitting a DMA buffer when the memory manager process determines that there is not enough room in the coprocessor-accessible memory location to page all needed memory resources. The Fadavi-Ardekani reference shows a memory partitioning method of splitting data buffer for a software application program (Column 5, lines 31-38). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multitasking processor system of Notenboom and Moore, as described above, further modifying the kernel of the operating system of Notenboom to have a memory partitioning scheme where data buffer is split to enable paging of variable length data buffers (Fadavi-Ardekani, Abstract) and it is also obvious to split a DMA buffer as well because it is a well known type of memory or buffer.

Claims 38-40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Notenboom et al (US 5,748,468) and Moore et al (US 5,437,017) as applied to claim 28 and further in view of Buckelew et al (US 5,864,512).

With respect to claim 38, all of the limitations of claim 28 have been addressed above. The Notenboom and Moore references do not teach a method further comprising: assigning a base address for a display surface wherein the display surface is allocated contiguously in coprocessor-readable memory; and delivering a task to the scheduler wherein processing the task will reassign the base address for a display surface. The Buckelew reference teaches a device for storing pixel information for displaying a graphics image on a display where pixels are stored at different addresses in memory (Column 1, line 55-column 2, line 4). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom with Moore, as described above, to further include Buckelew's device to buffer pixel images in different memory addresses, where displaying a different pixel corresponds to reassigning the base address for a display, to overcome the problems as described by Buckelew (Buckelew, column 1, lines 21-53), to increase efficiency and speed of graphics processing.

With respect to claim 39, all of the limitations of claims 28 and 38 have been addressed above. Refering to the above disussion of claim 38, it is inherent that the Buckelew reference teaches a method according to claim 38 wherein processing the task will reassign the base address for a display surface immediately because any

immediate action is obvious because it is a well-known desirability for graphics processing.

With respect to claim 40, all of the limitations of claims 28 and 38 have been addressed above. The Notenboom and Moore reference do not teach a method wherein processing the task will reassign the base address for a display surface upon the occurrence of a subsequent display synchronization period. The Buckelew reference teaches the reassigning of the base address for a display surface upon the occurrence of a subsequent display synchronization period (Column 13, lines 47-55). It would have been obvious at the time of the invention was made to a person having ordinary skill in the art to which said subject matter pertains to have modified the multiprocessor system of Notenboom with Moore, as described above, to further include Buckelew's device that reassigns the base address for a display surface upon the occurrence of a subsequent display synchronization period as suggested by Buckelew (Buckelew, Column 13, lines 47-55).

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. The Van Loo et al (US 5,016,161) reference teaches a page fault mechanism in a multi-tasking computer system. The Angle et al (US 5,761,506) and (US 6,272,516) references teaches a method for handling cache misses in a computer system. The Song (US 6,061,711) reference teaches a method of context saving and restoring in a multi-tasking computing system. The Tausheck (US 6,092,127) reference teaches a FIFO buffer that splits into three buffers.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kwok Wing Lee whose telephone number is (571) 270-3557. The examiner can normally be reached on Mon - Thu and alternate Fridays 7:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, David Robertson can be reached on (571) 272-4186. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/K. L./
Kwok W. Lee
11/15/2007

/David L. Robertson/
Supervisory Patent Examiner, Art Unit 4113